

**WHAT IS CLAIMED IS:**

1. A method of producing a binary code file comprising:

compiling a plurality of source code instructions; and

outputting a plurality of binary code instructions and compiler annotation.

2. The method as recited in Claim 1, wherein the compiler annotation enables

binary translation to be performed on the plurality of binary code instructions using a non-

heuristic approach.

3. The method as recited in Claim 1, wherein the compiler annotation describes

functional characteristics of the plurality of binary code instructions.

4. The method as recited in Claim 1, wherein the compiler annotation comprises

one or more records selected from a module identification (ID), a function ID, a split function

ID, a jump table ID, a function pointer initialization ID, a function address assignment ID, an

offset expression ID, a data in the text section ID, a volatile load ID, and an untouchable

region ID.

5. The method as recited in Claim 1, wherein the compiling the plurality of

source code instructions comprises:

examining the plurality of source code instructions;

reorganizing one or more of the plurality of source code instructions;

translating the plurality of source code instructions into the plurality of binary code

instructions;

reorganizing one or more of the plurality of binary code instructions; and

tracking and recording functional characteristics of the plurality of source code

instructions and of the plurality of binary code instructions.

6. The method as recited in Claim 1, wherein the plurality of binary code

instructions is an ELF format binary code file and the compiler annotation is an ELF section.

7. The compiler annotation created by the method of Claim 1.

8. A method of translating a source binary code file comprising:  
translating a plurality of source binary code instructions utilizing compiler annotation;  
and  
outputting a plurality of target binary code instructions.

9. The method as recited in Claim 8, wherein the compiler annotation enables the translating the plurality of source binary code instructions to be performed on the plurality of source binary code instructions using a non-heuristic approach.

10. The method as recited in Claim 8, wherein the compiler annotation describes functional characteristics of the plurality of binary code instructions.

11. The method as recited in Claim 8, wherein the compiler annotation comprises one or more records selected from a module identification (ID), a function ID, a split function ID, a jump table ID, a function pointer initialization ID, a function address assignment ID, an offset expression ID, a data in the text section ID, a volatile load ID, and an untouchable region ID.

12. The method as recited in Claim 8, wherein the translating the plurality of source binary code instructions comprises:  
utilizing the compiler annotation to partition the plurality of source binary code instructions into sections, functions and basic blocks; and  
building a control-flow graph utilizing the plurality of source binary code instructions and the compiler annotation.

13. The method as recited in Claim 8, wherein the plurality of source binary code instructions is an ELF format binary code file and the compiler annotation is an ELF section.

14. The method as recited in Claim 8, further comprising:  
outputting different compiler annotation.

15. The plurality of target binary code instructions and the different compiler annotation created by the method of Claim 14.

1 16. A binary code file comprising:  
 2 a plurality of binary code instructions; and  
 3 compiler annotation;  
 4 wherein the compiler annotation enables a binary translator to:  
 5 utilize the compiler annotation to partition the plurality of binary code  
 6 instructions into sections, functions and basic blocks; and  
 7 build a control-flow graph utilizing the plurality of binary code instructions  
 8 and the compiler annotation.

1 17. The binary code file as recited in Claim 16, wherein the compiler annotation  
 2 section enables binary translation to be performed on the plurality of binary code instructions  
 3 using a non-heuristic approach.

1 18. The binary code file as recited in Claim 16, wherein the compiler annotation  
 2 describes functional characteristics of the plurality of binary code instructions.

1 19. The binary code file as recited in Claim 16, wherein the compiler annotation  
 2 comprises one or more records selected from a module identification (ID), a function ID, a  
 3 split function ID, a jump table ID, a function pointer initialization ID, a function address  
 4 assignment ID, an offset expression ID, a data in the text section ID, a volatile load ID, and  
 5 an untouchable region ID.

1 20. The binary code file as recited in Claim 16, wherein the plurality of binary  
 2 code instructions and compiler annotation is an ELF format binary code file and the compiler  
 3 annotation is an ELF section.

1 21. An apparatus for producing a binary code file comprising:  
 2 means for compiling a plurality of source code instructions; and  
 3 means for outputting a plurality of binary code instructions and compiler annotation.

1 22. The apparatus as recited in Claim 21, wherein the compiler annotation enables  
 2 binary translation to be performed on the plurality of binary code instructions using a non-  
 3 heuristic approach.

1           23.    The apparatus as recited in Claim 21, wherein the compiler annotation  
2 describes functional characteristics of the plurality of binary code instructions.

1           24.    The apparatus as recited in Claim 21, wherein the compiler annotation  
2 comprises one or more records selected from a module identification (ID), a function ID, a  
3 split function ID, a jump table ID, a function pointer initialization ID, a function address  
4 assignment ID, an offset expression ID, a data in the text section ID, a volatile load ID, and  
5 an untouchable region ID.

1           25.    The apparatus as recited in Claim 21, wherein the means for compiling the  
2 plurality of source code instructions comprises:  
3           means for examining the plurality of source code instructions;  
4           means for reorganizing one or more of the plurality of source code instructions;  
5           means for translating the plurality of source code instructions into the plurality of  
6           binary code instructions;  
7           means for reorganizing one or more of the plurality of binary code instructions; and  
8           means for tracking and recording functional characteristics of the plurality of source  
9           code instructions and of the plurality of binary code instructions.

1           26.    An apparatus for translating a source binary code file comprising:  
2           means for translating a plurality of source binary code instructions utilizing compiler  
3           annotation; and  
4           means for outputting a plurality of target binary code instructions.

1           27.    The apparatus as recited in Claim 26, wherein the compiler annotation enables  
2 the translating the plurality of source binary code instructions to be performed on the plurality  
3 of source binary code instructions using a non-heuristic approach.

1           28.    The apparatus as recited in Claim 26, wherein the compiler annotation  
2 describes functional characteristics of the plurality of binary code instructions.

1           29.    The apparatus as recited in Claim 26, wherein the compiler annotation  
2 comprises one or more records selected from a module identification (ID), a function ID, a

split function ID, a jump table ID, a function pointer initialization ID, a function address assignment ID, an offset expression ID, a data in the text section ID, a volatile load ID, and an untouchable region ID.

30. The apparatus as recited in Claim 26, wherein the means for translating the plurality of source binary code instructions comprises:

means for utilizing the compiler annotation to partition the plurality of source binary code instructions into sections, functions and basic blocks; and  
means for building a control-flow graph utilizing the plurality of source binary code instructions and the compiler annotation.

31. An apparatus for producing a binary code file comprising:

a computer readable medium; and

instructions stored on the computer readable medium to:

compile a plurality of source code instructions; and

output a plurality of binary code instructions and compiler annotation.

32. The apparatus as recited in Claim 31, wherein the compiler annotation enables binary translation to be performed on the plurality of binary code instructions using a non-heuristic approach.

33. The apparatus as recited in Claim 31, wherein the compiler annotation describes functional characteristics of the plurality of binary code instructions.

34. The apparatus as recited in Claim 31, wherein the compiler annotation comprises one or more records selected from a module identification (ID), a function ID, a split function ID, a jump table ID, a function pointer initialization ID, a function address assignment ID, an offset expression ID, a data in the text section ID, a volatile load ID, and an untouchable region ID.

35. The apparatus as recited in Claim 31, wherein the instructions to compile the plurality of source code instructions comprises instructions to:

examine the plurality of source code instructions;

reorganize one or more of the plurality of source code instructions;

translate the plurality of source code instructions into the plurality of binary code instructions;  
reorganize one or more of the plurality of binary code instructions; and  
track and record functional characteristics of the plurality of source code instructions and of the plurality of binary code instructions.

36. An apparatus for translating a source binary code file comprising:  
a computer readable medium; and  
instructions stored on the computer readable medium to:  
translate a plurality of source binary code instructions utilizing compiler annotation; and  
output a plurality of target binary code instructions.

37. The apparatus as recited in Claim 36, wherein the compiler annotation enables the translating the plurality of source binary code instructions to be performed on the plurality of source binary code instructions using a non-heuristic approach.

38. The apparatus as recited in Claim 36, wherein the compiler annotation describes functional characteristics of the plurality of binary code instructions.

39. The apparatus as recited in Claim 36, wherein the compiler annotation comprises one or more records selected from a module identification (ID), a function ID, a split function ID, a jump table ID, a function pointer initialization ID, a function address assignment ID, an offset expression ID, a data in the text section ID, a volatile load ID, and an untouchable region ID.

40. The apparatus as recited in Claim 36, wherein the instructions to translate the plurality of source binary code instructions comprises instructions to:  
utilize the compiler annotation to partition the plurality of source binary code instructions into sections, functions and basic blocks; and  
build a control-flow graph utilizing the plurality of source binary code instructions and the compiler annotation.